**Working Code in laravel 9**

[**http://127.0.0.1:8000/login**](http://127.0.0.1:8000/login)

**http://127.0.0.1:8000/admin/login**

**1]** Create New Project

composer create-project laravel/laravel Larave9mauth

**2]** Install breeze package by using these four commands.

A] composer require laravel/breeze

B] php artisan breeze:install

C] npm install

D] npm run dev

**3]** Create Database and set name of database in .env file

**4]** Run this command to migrate

php artisan migrate

**5]** Now Authotication system had created.

**6]** Now create admin model and migration using this command

php artisan make:model Admin -m

Now model and migration are created successfully

**7]** Edit Admin migration file and copy this code from user migration file and change the table name as **admins**

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

/\*\*

\* Run the migrations.

\*

\* @return void

\*/

public function up()

{

Schema::create('admins', function (Blueprint $table) {

$table->id();

$table->string('name');

$table->string('email')->unique();

$table->timestamp('email\_verified\_at')->nullable();

$table->string('password');

$table->rememberToken();

$table->timestamps();

});

}

/\*\*

\* Reverse the migrations.

\*

\* @return void

\*/

public function down()

{

Schema::dropIfExists('admins');

}

};

**8]** Now, Edit the admin model and paste this code as user model and change the name as Admin and add protected $gurard = ‘admin’;

<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

use Laravel\Sanctum\HasApiTokens;

class Admin extends Authenticatable

{

use HasApiTokens, HasFactory, Notifiable;

protected $guard = 'admin';

/\*\*

\* The attributes that are mass assignable.

\*

\* @var array<int, string>

\*/

protected $fillable = [

'name',

'email',

'password',

];

/\*\*

\* The attributes that should be hidden for serialization.

\*

\* @var array<int, string>

\*/

protected $hidden = [

'password',

'remember\_token',

];

/\*\*

\* The attributes that should be cast.

\*

\* @var array<string, string>

\*/

protected $casts = [

'email\_verified\_at' => 'datetime',

];

}

**9]** Now, Open the config >> auth.php file to define the Guard and replace these code.

'guards' => [

'web' => [

'driver' => 'session',

'provider' => 'users',

],

'admin' => [

'driver' => 'session',

'provider' => 'admins',

],

],

and

'providers' => [

'users' => [

'driver' => 'eloquent',

'model' => App\Models\User::class,

],

'admins' => [

'driver' => 'eloquent',

'model' => App\Models\Admin::class,

],

// 'users' => [

// 'driver' => 'database',

// 'table' => 'users',

// ],

],

**10]** Now, migrate the admin table

php artisan migrate

**11]** Open app >> Http >> controller >> copy Auth folder >> and paste at same position named as **Adminauth**

**and** open all files into Adminauth folder and change the path from **Auth** to **Adminauth** as

**namespace App\Http\Controllers\Adminauth;**

**Now,** Open the first file and paste this code into **AuthenticatedSessionController** file

**AuthenticatedSessionController.php**

**<?php**

**namespace App\Http\Controllers\Adminauth;**

**use App\Http\Controllers\Controller;**

**use App\Http\Requests\Adminauth\LoginRequest;**

**use App\Providers\RouteServiceProvider;**

**use Illuminate\Http\Request;**

**use Illuminate\Support\Facades\Auth;**

**class AuthenticatedSessionController extends Controller**

**{**

**/\*\***

**\* Display the login view.**

**\***

**\* @return \Illuminate\View\View**

**\*/**

**public function create()**

**{**

**return view('admin.auth.login');**

**}**

**/\*\***

**\* Handle an incoming authentication request.**

**\***

**\* @param \App\Http\Requests\Auth\LoginRequest $request**

**\* @return \Illuminate\Http\RedirectResponse**

**\*/**

**public function store(LoginRequest $request)**

**{**

**$request->authenticate();**

**$request->session()->regenerate();**

**return redirect()->intended(RouteServiceProvider::ADMIN\_HOME);**

**}**

**/\*\***

**\* Destroy an authenticated session.**

**\***

**\* @param \Illuminate\Http\Request $request**

**\* @return \Illuminate\Http\RedirectResponse**

**\*/**

**public function destroy(Request $request)**

**{**

**// Auth::guard('admin')->logout();**

**// $request->session()->invalidate();**

**// $request->session()->regenerateToken();**

**// return redirect('/');**

**Auth::guard('admin')->logout();**

**return redirect()->route('admin.login');**

**}**

**}**

**Now,** Open the second file **ConfirmablePasswordController** file, and change the code.

<?php

namespace App\Http\Controllers\Adminauth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Illuminate\Validation\ValidationException;

class ConfirmablePasswordController extends Controller

{

/\*\*

\* Show the confirm password view.

\*

\* @return \Illuminate\View\View

\*/

public function show()

{

return view('admin.auth.confirm-password');

}

/\*\*

\* Confirm the user's password.

\*

\* @param \Illuminate\Http\Request $request

\* @return mixed

\*/

public function store(Request $request)

{

if (! Auth::guard('admin')->validate([

'email' => $request->user()->email,

'password' => $request->password,

])) {

throw ValidationException::withMessages([

'password' => \_\_('auth.password'),

]);

}

$request->session()->put('auth.password\_confirmed\_at', time());

return redirect()->intended(RouteServiceProvider::ADMIN\_HOME);

}

}

**Now,** Open the third file and paste this code into **EmailVerificationNotificationController** file

<?php

namespace App\Http\Controllers\Adminauth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Http\Request;

class EmailVerificationNotificationController extends Controller

{

/\*\*

\* Send a new email verification notification.

\*

\* @param \Illuminate\Http\Request $request

\* @return \Illuminate\Http\RedirectResponse

\*/

public function store(Request $request)

{

if ($request->user()->hasVerifiedEmail()) {

return redirect()->intended(RouteServiceProvider::ADMIN\_HOME);

}

$request->user()->sendEmailVerificationNotification();

return back()->with('status', 'verification-link-sent');

}

}

**Now,** Open the fourth file and paste this code into **EmailVerificationPromptController** file

<?php

namespace App\Http\Controllers\Adminauth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Http\Request;

class EmailVerificationPromptController extends Controller

{

/\*\*

\* Display the email verification prompt.

\*

\* @param \Illuminate\Http\Request $request

\* @return mixed

\*/

public function \_\_invoke(Request $request)

{

return $request->user()->hasVerifiedEmail()

? redirect()->intended(RouteServiceProvider::ADMIN\_HOME)

: view('auth.verify-email');

}

}

**Now,** Open the fifth file and paste this code into **NewPasswordController** file

<?php

namespace App\Http\Controllers\Adminauth;

use App\Http\Controllers\Controller;

use Illuminate\Auth\Events\PasswordReset;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Hash;

use Illuminate\Support\Facades\Password;

use Illuminate\Support\Str;

use Illuminate\Validation\Rules;

class NewPasswordController extends Controller

{

/\*\*

\* Display the password reset view.

\*

\* @param \Illuminate\Http\Request $request

\* @return \Illuminate\View\View

\*/

public function create(Request $request)

{

return view('admin.auth.reset-password', ['request' => $request]);

}

/\*\*

\* Handle an incoming new password request.

\*

\* @param \Illuminate\Http\Request $request

\* @return \Illuminate\Http\RedirectResponse

\*

\* @throws \Illuminate\Validation\ValidationException

\*/

public function store(Request $request)

{

$request->validate([

'token' => ['required'],

'email' => ['required', 'email'],

'password' => ['required', 'confirmed', Rules\Password::defaults()],

]);

// Here we will attempt to reset the user's password. If it is successful we

// will update the password on an actual user model and persist it to the

// database. Otherwise we will parse the error and return the response.

$status = Password::reset(

$request->only('email', 'password', 'password\_confirmation', 'token'),

function ($user) use ($request) {

$user->forceFill([

'password' => Hash::make($request->password),

'remember\_token' => Str::random(60),

])->save();

event(new PasswordReset($user));

}

);

// If the password was successfully reset, we will redirect the user back to

// the application's home authenticated view. If there is an error we can

// redirect them back to where they came from with their error message.

return $status == Password::PASSWORD\_RESET

? redirect()->route('login')->with('status', \_\_($status))

: back()->withInput($request->only('email'))

->withErrors(['email' => \_\_($status)]);

}

}

**Now,** Open the six file and paste this code into **PasswordResetLinkController** file

<?php

namespace App\Http\Controllers\Adminauth;

use App\Http\Controllers\Controller;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Password;

class PasswordResetLinkController extends Controller

{

/\*\*

\* Display the password reset link request view.

\*

\* @return \Illuminate\View\View

\*/

public function create()

{

return view('admin.auth.forgot-password');

}

/\*\*

\* Handle an incoming password reset link request.

\*

\* @param \Illuminate\Http\Request $request

\* @return \Illuminate\Http\RedirectResponse

\*

\* @throws \Illuminate\Validation\ValidationException

\*/

public function store(Request $request)

{

$request->validate([

'email' => ['required', 'email'],

]);

// We will send the password reset link to this user. Once we have attempted

// to send the link, we will examine the response then see the message we

// need to show to the user. Finally, we'll send out a proper response.

$status = Password::sendResetLink(

$request->only('email')

);

return $status == Password::RESET\_LINK\_SENT

? back()->with('status', \_\_($status))

: back()->withInput($request->only('email'))

->withErrors(['email' => \_\_($status)]);

}

}

**Now,** Open the seventh file and paste this code into **RegisteredUserController** file

<?php

namespace App\Http\Controllers\Adminauth;

use App\Http\Controllers\Controller;

use App\Models\Admin;

use App\Providers\RouteServiceProvider;

use Illuminate\Auth\Events\Registered;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\Hash;

use Illuminate\Validation\Rules;

class RegisteredUserController extends Controller

{

/\*\*

\* Display the registration view.

\*

\* @return \Illuminate\View\View

\*/

public function create()

{

return view('admin.auth.register');

}

/\*\*

\* Handle an incoming registration request.

\*

\* @param \Illuminate\Http\Request $request

\* @return \Illuminate\Http\RedirectResponse

\*

\* @throws \Illuminate\Validation\ValidationException

\*/

public function store(Request $request)

{

$request->validate([

'name' => ['required', 'string', 'max:255'],

'email' => ['required', 'string', 'email', 'max:255', 'unique:admins'],

'password' => ['required', 'confirmed', Rules\Password::defaults()],

]);

$user = Admin::create([

'name' => $request->name,

'email' => $request->email,

'password' => Hash::make($request->password),

]);

event(new Registered($user));

Auth::guard('admin')->login($user);

return redirect(RouteServiceProvider::ADMIN\_HOME);

}

}

**Now,** Open the eight file and paste this code into **VerifyEmailController** file

<?php

namespace App\Http\Controllers\Adminauth;

use App\Http\Controllers\Controller;

use App\Providers\RouteServiceProvider;

use Illuminate\Auth\Events\Verified;

use Illuminate\Foundation\Auth\EmailVerificationRequest;

class VerifyEmailController extends Controller

{

/\*\*

\* Mark the authenticated user's email address as verified.

\*

\* @param \Illuminate\Foundation\Auth\EmailVerificationRequest $request

\* @return \Illuminate\Http\RedirectResponse

\*/

public function \_\_invoke(EmailVerificationRequest $request)

{

if ($request->user()->hasVerifiedEmail()) {

return redirect()->intended(RouteServiceProvider::ADMIN\_HOME.'?verified=1');

}

if ($request->user()->markEmailAsVerified()) {

event(new Verified($request->user()));

}

return redirect()->intended(RouteServiceProvider::ADMIN\_HOME.'?verified=1');

}

}

**12]** Now, Open the App >> providers >> RouteServiceProvider.php file and add this code

public const HOME = '/dashboard';

public const ADMIN\_HOME = '/admin/dashboard';

**13]** Now go to the App >> Http >> Requests >> Copy the Auth folder >> and paste at the same location named as **Adminauth** and edit LoginRequest.php file and paste this code into this file.

<?php

namespace App\Http\Requests\Adminauth;

use Illuminate\Auth\Events\Lockout;

use Illuminate\Foundation\Http\FormRequest;

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\RateLimiter;

use Illuminate\Support\Str;

use Illuminate\Validation\ValidationException;

class LoginRequest extends FormRequest

{

/\*\*

\* Determine if the user is authorized to make this request.

\*

\* @return bool

\*/

public function authorize()

{

return true;

}

/\*\*

\* Get the validation rules that apply to the request.

\*

\* @return array

\*/

public function rules()

{

return [

'email' => ['required', 'string', 'email'],

'password' => ['required', 'string'],

];

}

/\*\*

\* Attempt to authenticate the request's credentials.

\*

\* @return void

\*

\* @throws \Illuminate\Validation\ValidationException

\*/

public function authenticate()

{

$this->ensureIsNotRateLimited();

if (! Auth::guard('admin')->attempt($this->only('email', 'password'), $this->boolean('remember'))) {

RateLimiter::hit($this->throttleKey());

throw ValidationException::withMessages([

'email' => trans('auth.failed'),

]);

}

RateLimiter::clear($this->throttleKey());

}

/\*\*

\* Ensure the login request is not rate limited.

\*

\* @return void

\*

\* @throws \Illuminate\Validation\ValidationException

\*/

public function ensureIsNotRateLimited()

{

if (! RateLimiter::tooManyAttempts($this->throttleKey(), 5)) {

return;

}

event(new Lockout($this));

$seconds = RateLimiter::availableIn($this->throttleKey());

throw ValidationException::withMessages([

'email' => trans('auth.throttle', [

'seconds' => $seconds,

'minutes' => ceil($seconds / 60),

]),

]);

}

/\*\*

\* Get the rate limiting throttle key for the request.

\*

\* @return string

\*/

public function throttleKey()

{

return Str::transliterate(Str::lower($this->input('email')).'|'.$this->ip());

}

}

**14]** Now, open the routes >> web.php file and past this code for admin dashboard

Route::get('/admin/dashboard', function () {

return view('admin.dashboard');

})->middleware(['auth:admin'])->name('admin.dashboard');

require \_\_DIR\_\_.'/adminauth.php';

**15]** Now, go to the routes folder >> copy the auth.php file >> past at the same location named as adminauth.php and paste this code into adminauth.php file

<?php

use App\Http\Controllers\Adminauth\AuthenticatedSessionController;

use App\Http\Controllers\Adminauth\ConfirmablePasswordController;

use App\Http\Controllers\Adminauth\EmailVerificationNotificationController;

use App\Http\Controllers\Adminauth\EmailVerificationPromptController;

use App\Http\Controllers\Adminauth\NewPasswordController;

use App\Http\Controllers\Adminauth\PasswordResetLinkController;

use App\Http\Controllers\Adminauth\RegisteredUserController;

use App\Http\Controllers\Adminauth\VerifyEmailController;

use Illuminate\Support\Facades\Route;

Route::group(['middlewar' => ['guest:admin'],'prefix' => 'admin','as'=>'admin.'],function(){

//Route::prefix('admin')->group(function () {

Route::get('/register', [RegisteredUserController::class, 'create'])

->name('register');

Route::post('/register', [RegisteredUserController::class, 'store']);

Route::get('/login', [AuthenticatedSessionController::class, 'create'])

->name('login');

Route::post('/login', [AuthenticatedSessionController::class, 'store']);

Route::get('forgot-password', [PasswordResetLinkController::class, 'create'])

->name('password.request');

Route::post('forgot-password', [PasswordResetLinkController::class, 'store'])

->name('password.email');

Route::get('reset-password/{token}', [NewPasswordController::class, 'create'])

->name('password.reset');

Route::post('reset-password', [NewPasswordController::class, 'store'])

->name('password.update');

});

Route::group(['middlewar' => ['guest:admin'],'prefix' => 'admin','as'=>'admin.'],function(){

//Route::group(['middleware' => ['auth:admin'],'prefix'=>'admin','as'=>'admin'],function(){

Route::get('verify-email', [EmailVerificationPromptController::class, '\_\_invoke'])

->name('verification.notice');

Route::get('verify-email/{id}/{hash}', [VerifyEmailController::class, '\_\_invoke'])

->middleware(['signed', 'throttle:6,1'])

->name('verification.verify');

Route::post('email/verification-notification', [EmailVerificationNotificationController::class, 'store'])

->middleware('throttle:6,1')

->name('verification.send');

Route::get('confirm-password', [ConfirmablePasswordController::class, 'show'])

->name('password.confirm');

Route::post('confirm-password', [ConfirmablePasswordController::class, 'store']);

//Route::post('logout', [AuthenticatedSessionController::class, 'destroy'])

->name('logout');

Route::get('/logout',[App\Http\Controller\AdminController::class,'logout'])->name('admin.logout');

});

**16]** Now, Go to the resources >> view >> make admin folder >> copy auth, layout folder into admin folder and >> copy dashboard.blade.php file >> and past into admin folder.

**dashboard.blade.php**

<x-admin-layout>

<x-slot name="header">

<h2 class="font-semibold text-xl text-gray-800 leading-tight">

Admin {{ \_\_('Dashboard') }}

</h2>

</x-slot>

<div class="py-12">

<div class="max-w-7xl mx-auto sm:px-6 lg:px-8">

<div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">

<div class="p-6 bg-white border-b border-gray-200">

You're logged in!

</div>

</div>

</div>

</div>

</x-admin-layout>

**17]** Add **admin. Into route** Each and every file of auth folder like as

<form method="POST" action="{{ route('**admin.**password.email') }}">

And change layout path as

<x-admin-guest-layout>

**18]** Finally, open the App >> View >> components >> copy the AppLayout.php >> past at the same location named as AdminLayout.php and past the code like as

<?php

namespace App\View\Components;

use Illuminate\View\Component;

class AdminLayout extends Component

{

/\*\*

\* Get the view / contents that represents the component.

\*

\* @return \Illuminate\View\View

\*/

public function render()

{

return view('admin.layouts.app');

}

}

And open the App >> View >> components >> copy the GuestLayout.php >> past at the same location named as AdminGuestLayout.php and past the code like as

<?php

namespace App\View\Components;

use Illuminate\View\Component;

class AdminGuestLayout extends Component

{

/\*\*

\* Get the view / contents that represents the component.

\*

\* @return \Illuminate\View\View

\*/

public function render()

{

return view('admin.layouts.guest');

}

}

**19]** Go to the App >> Middleware >> open the RedirectifAuthenticated.php file and past this code as

<?php

namespace App\Http\Middleware;

use App\Providers\RouteServiceProvider;

use Closure;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

class RedirectIfAuthenticated

{

/\*\*

\* Handle an incoming request.

\*

\* @param \Illuminate\Http\Request $request

\* @param \Closure(\Illuminate\Http\Request): (\Illuminate\Http\Response|\Illuminate\Http\RedirectResponse) $next

\* @param string|null ...$guards

\* @return \Illuminate\Http\Response|\Illuminate\Http\RedirectResponse

\*/

public function handle(Request $request, Closure $next, ...$guards)

{

$guards = empty($guards) ? [null] : $guards;

foreach ($guards as $guard) {

if ($guard == 'admin' && Auth::guard($guard)->check()) {

return redirect(RouteServiceProvider::ADMIN\_HOME);

}

if (Auth::guard($guard)->check()) {

return redirect(RouteServiceProvider::HOME);

}

}

return $next($request);

}

}